Research in Computational Astrobiology
Final Technical Report for Contract NCC2-5453

Galina Chaban[a], Richard Jaffe[b], Shoudan Liang[c], Michael H. New[d], Andrew Pohorille[d,e], and Michael A. Wilson[d,e,f]

[a]MS T27B-1, NASA Ames Research Center, Moffett Field, CA 94035

[b]MS 230-3, NASA Ames Research Center, Moffett Field, CA 94035

[c]MS 229-1, NASA Ames Research Center, Moffett Field, CA 94035

[d]MS 239-4, NASA Ames Research Center, Moffett Field, CA 94035

[e]Dept. Pharmaceutical Chemistry, University of California, San Francisco, 94143

[f]Principal Investigator

Period of Performance: August 1, 2001 – July 31, 2002

Inventions: There are no patents or inventions arising from this research project.

Abstract

We present results from several projects in the new field of computational astrobiology, which is devoted to advancing our understanding of the origin, evolution and distribution of life in the Universe using theoretical and computational tools. We have developed a procedure for calculating long-range effects in molecular dynamics using a plane wave expansion of the electrostatic potential. This method is expected to be highly efficient for simulating biological systems on massively parallel supercomputers. We have perform genomics analysis on a family of actin binding proteins. We have performed quantum mechanical calculations on carbon nanotubes and nucleic acids, which simulations will allow us to investigate possible sources of organic material on the early earth. Finally, we have developed a model of protobiological chemistry using neural networks.

Introduction

The goal of this research has been to pursue research in computational astrobiology through a series of projects, suitable for advanced undergraduate students, that address several issues identified in the recent NASA Astrobiology Roadmap. Computational Astrobiology uses computational and theoretical techniques to advance our understanding of the origin, evolution and distribution of life in the Universe. These problems are approached from several different points of view, ranging from the molecular and cellular level to the ecological and biosphere level. This requires exploiting information from not only the biological sciences, but also chemistry, geology, paleontology, and planetary and atmospheric sciences.

Similarly, the goals of computational astrobiology cannot be accomplished using a single area of computer science but, instead, involve creative integration of several traditionally separate disciplines: biomodeling and biosimulations, bioinformatics, and complex systems science.

Long-range Effects in Molecular Dynamics Simulations. To assist Ames' efforts in developing fast code for molecular-level simulation of biological systems on massively parallel supercomputers, we have developed a modified, highly efficient, state-of-the-art, plane-wave expansion code to treat long range effects in molecular dynamics simulations. The newly developed code will be applied to the study of two outstanding problems in astrobiology: (i) understanding the structure and mechanism of action of the first proteins evolved from random sequences by in vitro selection; and (ii) designing simple membrane proteins capable of transporting material across cell walls, utilizing energy captured from the environment and transmitting signals from the environment into the prebiotic cell. This work will truly advance both the state-of-the-art of research in astrobiology and the connection between astrobiology and information technology at Ames.

We have previously developed a multipole expansion code for isolated systems. The plane-wave expansions of the electrostatic potential appears to be suitable for parallel computation as it requires less communication than the expansion in spherical harmonics.

Electronic Structure Calculations on Nucleic Acid Bases. The project has involved ab initio studies of electronic structure and spectral properties of such biological building blocks as DNA bases. Structure and vibrational spectra of different tautomers of DNA bases and their complexes with water were computed in order to understand the effects of conformational structure and hydrogen bonding on vibrational frequencies/intensities. These results should be helpful in possible identification of these molecules in space and/or interpretation of laboratory measured infrared spectra of biological systems. In addition, electronic excitation spectra and excited state proton-transfer reactions in DNA bases were investigated using multi-configurational wave functions. Tautomerization processes, that take place as a result of photoinduced proton transfer reactions, are believed to be a first step in mutation of DNA. It is therefore important to study the effect of electronic excitation on the proton transfer process.

Quantum Mechanical Studies of Modified Carbon Naontubes. Carbon nanotubes are being evaluated for use as probes for Atomic Force Microscopy (AFM). It is already known that they have excellent mechanical properties like stiffness and resilience. So they can make durable AFM probes for metrology and imaging applications. Except for their ends, they also are chemically inert. In order to make chemically selective probes, there is interest in studying substitution reactions for nanotube tips. If that chemistry can be controlled, scientists should be able design highly selective single-molecule probes. We have investigated the properties of carbon nanotubes with chemically modified ends using quantum chemical calculations.

The calculations were carried out for fragments of nanotubes with carboxyl, amide and

ester groups attached. We have determined bond strengths for these groups and derived force fields for molecular dynamics calculations of functionalized nanotubes in aqueous solution. Similar quantum chemistry calculations have been carried out for other modifications of nanotubes, so the proposed computational methodology has been validated previously.

Metabolic Profiling by Multidimensional NMR Recently-developed high-throughput techniques such as DNA micro-arrays have made it possible to simultaneously monitor the expression levels of all the genes in a cell. However, elucidating cellular functions with the large body of data thus generated has so far been difficult, mainly because there are no corresponding quantitative measures of cellular functions to parallel the gene expression data. Knowing which genes or groups of genes have similar expression patterns is interesting, but such knowledge would be much more useful if it could be correlated with simultaneous measurements of the many metabolic processes that these expressed genes may regulate. If these data can be obtained, they could potentially unearth many as-yet-unrecognized roles for genes whose functions are incompletely understood, and illuminate the feedback mechanisms and pathways responsible for controlling their expression levels.

To this end, we are developing multidimensional nuclear magnetic resonance (NMR) and associated computational techniques that will detect the level of as many metabolites as possible in vivo in a high-throughput fashion. Because small molecule metabolites in a cell are a direct and quantitative characterization of its metabolic functions, the data obtained with our method complements the DNA micro-array gene expression data and assist efforts to interpret them. As a first step, we have developed algorithms and computer code in order to quantify the positions and the magnitudes of peaks in a two-dimensional NMR spectrum.

Structural Homology Prediction and MD Modeling to Find GdETmod Structure. The focus of this project was to deduce structure-function relationships for members of the Tropomodulin (Tmod) family of actin-filament capping proteins, using computational and evolutionary approaches. The Tmods make up a small well-conserved protein family, the members of which show no significant sequence similarity to other known proteins. Functionally, every member tested of the Tmod family is able to bind at least some tropomyosin isoforms in in vitro assays as well as in vivo. Tmods display capping activity at the slow-growing 'pointed' end of pure or tropomyosin-coated actin filaments. The Tmods are the only proteins known to perform both these biochemical functions in one protein, and until recently were the only proteins known to cap actin filaments at the slow-growing ends. Structurally, the Tmods are divided in two subfamilies, the 40kD tropomodulins and the 60-70kD leiomodins (Lmods). Four 40kD Tmod isoforms have been found in vertebrates, and all four isoforms are conserved between rats, mice, and humans; to date, some but not all of these isoforms have been identified in birds, amphibians, and fish.

Results

1 Long-range Effects in Molecular Dynamics

(points of contact: Andrew Pohorille and Michael Wilson)

Teena Gerhardt, Stanford University

1.1 Introduction

This document describes an implementation of the Plane Wave Fast Multipole Method developed by Eric Darve. We first outline the theoretical foundation of the method prior to delving into the implementation details.

1.2 The Method

This new formulation, the Plane Wave Fast Multipole Method (PW-FMM) is based on the expansion

$$\frac{1}{|r|} = \frac{1}{\pi} \int_{\chi=0}^{\infty} \int_{\phi=0}^{2\pi} e^{-\chi z} e^{i\chi(x\cos\phi + y\sin\phi)} d\chi d\phi$$

where $r = (x, y, z)$. From this expansion, we develop a method of computing multipole expansions more efficiently than previous methods.

The PW-FMM is divided into several steps. These are described below. We first introduce the following notation. Let $\sigma = (\chi, \phi)$ where χ and ϕ are continuous parameters. The x_i denote the coordinates of the particle i. The z_k indicate the center coordinates of the cluster C_k.

1. Initialization- We initialize the multipole coefficient f_{C_k} at a discrete point σ.

$$f_{C_k}(\sigma) = \sum_{x_i \in C_k} e^{-\chi(x_i - z_k)_z} e^{i\chi((x_i - z_k)_x \cos\phi + (x_i - z_k)_y \sin\phi))}. \quad (1)$$

This shifts the information for each particle to the center of the cluster in which it is contained. As the particles themselves are only stored in the lowest level of the tree, in the next step we shift the particle information up the tree.

2. Multipole to Multipole- This step shifts the multipole information from the child to its parent and for each parent sums this information over all of its children.

$$f_C(\sigma) = \sum_{i=1}^{8} f_{C_i}(\sigma) e^{-\chi(z_i - z)_z} e^{i\chi((z_i - z)_x \cos\phi + (z_i - z)_y \sin\phi))}. \quad (2)$$

3. Multipole to Local- This step finds the local coefficients for each cluster by transferring

informations from the surrounding clusters. This is done through the use of six interaction lists, one each in the +/- x, y, z directions. These interaction lists indicate clusters which are well-separated from the cluster in question, yet whose parents are not well-separated from the parent of the cluster. The following calculation is performed in each of the six directions for all r such that C_r appears in the interaction list of C_k.

$$g_{C_k}(\sigma) = \sum_r f_{c_r}(\sigma) T_{l,\sigma}(z_k - z_r) \tag{3}$$

We now elaborate on the construction of the transfer function $T_{l,\sigma_q}(z_k - z_r)$. In order to construct this function we first identify the least number of frequencies l needed for $T_{l,\sigma}(z_k - z_r)$ for a given accuracy ϵ. We then Fourier transform the following function

$$e^{-\chi(z_k - z_r)_z} e^{i\chi((z_k - z_r)_x \cos\phi + (z_k - z_r)_y \sin\phi)} \tag{4}$$

retain only the frequencies through l, and inverse Fourier transform our function to obtain our smoothed transfer function $T_{l,\sigma}(z_k - z_r)$. This transfer function accommodates the optimal number of sample points for a given error ϵ. Once the transfer function has been constructed, the local coefficients are calculated by applying equation 3.

4. Local to Local - The information is then shifted from the parents back down to their children.

$$g_{C_i}(\sigma) += g_C(\sigma) e^{-\chi(z - z_i)_z} e^{i\chi((z - z_i)_x \cos\phi + (z - z_i)_y \sin\phi))}. \tag{5}$$

This step moves down the tree, incorporating the transfer information from the parents in to the local coefficients of their children.

5. Integration - We next integrate to find the force at each particle

$$v_j = \int_{\chi=0}^{\infty} \int_{\phi=0}^{2\pi} g_{C_k}(\chi, \phi) e^{-\chi(z_k - x_j)_z} e^{i\chi((z_k - x_j)_x \cos\phi + (z_k - x_j)_y \sin\phi))} d\chi d\phi. \tag{6}$$

This integration is done by using discrete points and associated weights. For each discrete point (χ_n, ϕ_n) we have an associated weight ω_{χ_n, ϕ_n}. Then

$$v_j \approx \sum_n \omega_{\chi_n, \phi_n} g_{C_k}(\chi_n, \phi_n) e^{-\chi_n(z_k - x_j)_z} e^{i\chi_n((z_k - x_j)_x \cos\phi_n + (z_k - x_j)_y \sin\phi_n))} \tag{7}$$

6. Aggregation - As a final step we sum along all six directions +/- x, y, z and then incorporate the direct interactions between close particles to calculate the total force on each particle.

1.3 Implementation

We first describe the data structures utilized before giving a detailed account of the implementation of the method.

1.3.1 Data Structures

Tree Structure. We support a nonadaptive tree structure, stored as a 4-dimensional dynamically allocated array of pointers to cluster structures. The first index in the array indicates the level of the cluster within the tree structure, and the remaining three indices indicate the x, y, and z indices of the cluster respectively. This 4-D array provides a convenient notation. Note, for instance, that the parent of the cluster pointed to by boxes[n][x][y][y] is pointed to by boxes[n-1][x/2][y/2][z/2] (integer division). This 4-D array is wrapped in a higher level tree structure which also contains the array of particles for the tree and parameters describing the size and depth of the tree. The discretization points and weights used throughout the calculation are also stored in the tree structure. Additionally, this tree structure stores arrays holding values of sines and cosines respectively that are calculated in the initialization stages of the method and then reused in integration. Storing these values instead of recalculating them proves to increase the efficiency of the method.
Cluster Structure. Each cluster stores its own center coordinates, the expansion data (multipole and local coefficients) at the discretization points, and six interaction lists, in the $+/-x, y, z$ directions. These interaction lists are stored as statically allocated 2-D array structures which contain a pointer to the cluster interacted with and indices indicating which periodic instance of this cluster is being considered. The first index of the 2-D array indicates the direction of interaction for the list. In each direction these lists contain elements which are well-separated from the cluster in question yet whose parents are not well-separated from the cluster's parents. The local and multipole coefficients at each discretization point are also stored in the cluster structure. Additionally, the cluster structure contains an array of pointers to the particles contained in that cluster. Finally there are two arrays stored in each cluster that contain values used in the multipole to local step. These values are independent of the particle locations and charges, and thus can be computed in preprocessing stages and stored for later retrieval.

1.3.2 Details of the PW-FMM

Preprocessing. The preprocessing portion of the code is executed only once for each size system. In this section the tree structure itself is built, the discretization points are created, coefficients are dynamically allocated and initialized, and values are computed and stored that will later be used in the multipole to local stage. We now elaborate on a few of these steps.

Discretization points are generated as follows. The discretization of χ is given by an optimal quadrature point routine which uses an iterative Newton-Raphson solver. This routine provides the values and weights for χ. We then generate a uniform discretization of ϕ by finding the minimum number of discretization points needed for an accuracy ϵ. This is done by sampling many values of ϕ, calculating

$$e^{-\chi z} e^{i\chi(x\cos\phi + y\sin\phi)}$$

for these values of ϕ and χ and then fast Fourier transforming our vector. We then use this transformed vector to truncate at the given error epsilon, thereby determining the number of discretization points necessary for a given value of χ on a given level. This is done for all χ on all levels and the maximum of these values is chosen. This will give us a good representation of our function at all levels.

Many of the values needed for the transfer, or multipole to local, step of the method can be calculated in preprocessing and stored for later retrieval and use. Thus in preprocessing stages we calculate

$$e^{-\chi(z_k-z_r)_z} e^{i\chi((z_k-z_r)_x \cos\phi + (z_k-z_r)_y \sin\phi)}, \tag{8}$$

storing the result in its real and imaginary parts.

Direct Interactions. Following the preprocessing stages, for each cluster the direct interactions are calculated between each particle with the other particles in the cluster, and also between it and the particles in the immediately surrounding clusters (i.e. those clusters that are not well-separated from the cluster in question).

FMM. The FMM portion of the method is separated into stages as introduced earlier. The decomposition of the implementation follows the above outline. The FMM as coded implements the equations for each discretization point, then uses these points and the associated weights in the final integration step to generate the forces and potentials. We now elaborate on a few noteworthy details of the implementation. The computations in the FMM are only executed for the first half of the discretization points of ϕ. Since we are using a uniform discretization from 0 to 2π we are able to calculate only for values of ϕ between 0 and π, inferring the remaining results from this data. Additionally, in order to avoid recalculation we store many of the values calculated when initializing the multipole coefficients for reuse in the integration step. Finally, noting similarities in calculations between the positive and negative interactions in a given direction allows us to reduce the number of total calculations by exploiting these similarities.

Remark 1 *For explicit details of implementation, see the README file in the FMM directory and the extensive commenting in fmm.c.*

1.3.3 Usage

The program is executed by running fmm. The user is then prompted to enter the number of separate input files to be considered (note all input files must have the same number of particles in the same size boxes), the names of these input files, the name of the file to which output should be written, the number of levels in the tree structure, and the error epsilon for the forces to be calculated.

The program assumes a particular structure of input file. See the README file in the FMM directory for explicit details on this structure.

1.3.4 Results

Timings were done on an SGI with a 225 MHz Mips R10000 processor. While efforts were made to increase the speed of the PW-FMM, it remains quite slow due to the large number of computations, particularly sines, cosines, and exponentiations, that are required. For a system of 8000 water molecules (24000 atoms)and an absolute accuracy of .001, the calculation after preprocessing takes 41 seconds, 19 of which is the direct interactions and 22 of which is the PW-FMM itself. For a system of 27,000 water molecules, the calculation after preprocessing takes 275 seconds, 216 seconds accounted for by direct interactions and 59 accounted for by the PW-FMM. Note that the PW-FMM section of the code is approximately linear with the number of particles. While these results may be slow, analysis of where in the calculation the most time is consumed reveals that most time is used in the initialization of multipole coefficients in each cluster and in the integration of each cluster (8 seconds, and 8 seconds respectively for the 8000 molecule case). The advantage of time being consumed in these stages is that these steps are easily parallelizable.

The error in forces produced by this implementation can be reduced as low as 10^{-12}.

2 Electronic Structure Calculations on Nucleic Acid Bases

(point of contact: Galina Chaban)

Latasha Salter, Jackson State University

Electronic structure and spectral properties of tautomers of adenine were studied. Structure and vibrational spectra were computed in an effort to better understand the effects of conformational structure and hydrogen bonding on vibrational frequencies and intensities. The results should be helpful in possible identification of these molecules in space or interpretation of laboratory measured infrared spectra of biological systems. In addition, electronic excitation spectra and excited state proton-transfer reactions in adenine were investigated using multi-configurational wave functions. Tautomerization processes, that occur as a result of photoinduced proton transfer reactions, are believed to be the first step in the mutation of DNA. Therefore, it is important to study the effect of the electronic excitation on the proton-transfer process.

The Multi-Configurational Self-Consistent Field was utilized in this study. MCSCF is one of the fundamental methods in electronic structure theory that accounts for non-dynamic correlation. It is, especially, important for correct description of chemical reactions that involve multiple bond breaking. Part of orbitals in a MCSCF wavefunction can be fixed to be doubly occupied in all configurations. The orbitals are known as core because they are inactive. Active orbitals are allowed to have variable occupation numbers. Virtual orbitals

are those that are always empty. In this study, an active space of eight electrons and eight orbitals were included in correlation. This is denoted as MCSCF(8,8). All eight electrons and eight orbitals included in the active space have pi character. In this study, MCSCF method was applied in order to evaluate excited states in addition to the ground state. Dunning-Hay double-zeta plus polarization basis set (DZP) was utilized for geometry optimizations. In addition, multi-configurational second order perturbation theory (MCQDPT2) was used to improve the energetics. All calculations were done utilizing the General Atomic and Molecular Structure System (GAMESS). Four different tautomers of the adenine molecule were studied. Each has the hydrogen atom connected to the different nitrogen atom of the adenine rings. Both, the ground and first excited electrons states were considered. After geometrical structures of the tautomers were optimized, second derivative Hessian matrices were calculated and vibrational frequencies were obtained.

All MCQDPT2 results have not been, yet, obtained, as of now 3-adenine is more stable on the ground and excited states for this theory. The suspected transition state going from tautomer 9 to 3 has been located. No hypothesis has been formulated, as to why, 3-adenine is more stable than 7-adenine on the ground and excited stated when using MCQDPT2. For MCSCF calculations, 9-adenine was the most stable on the ground state and the second most stable on the excited state. Also, 1-adenine was the least stable on, both, the ground and excited states. To our dismay, 3-adenine was the most stable on the excited state using this theory.

In addition relative energetics, we plan to evaluate barrier heights for proton-transfer reactions between the different tautomers. Transfer between 9-adenine and 3-adenine is of special interest, since the hydrogen atom can transfer directly. It would be of great interest to know whether the barrier height is lower on the excited state compared to the ground state. It would mean that proton-transfer reactions can proceed more readily on the excited state than on the ground state.

3 Quantum Mechanical Studies of Modified Carbon Naontubes

(point of contact: Richard Jaffe)

Tomekia Simeon, Jackson State University

Carbon nanotubes contain a range of properties that make them well suited for use as probe tips in applications such as Atomic Force Microscopy. Initially, silicon probes were used for AFM. Further studies indicated that these tips place significant constraints on potential lateral resolution and the pyramidal shapes of the probe restricts the ability of these tips to access narrow and deep crevices. By attaching Multiwall Carbon Nanotubes (MWNTs) to the ends of the Si tips the cylindrical geometry of the tips provided imaging with an excellent resolution. In addition, carbon nanotubes elasticity buckle above a critical force. This buckling is relevant because it prevents damage to delicate organic and biological

samples. Thus the preparation of a wide range of functionalized nanotubes should be possible to use in various imaging applications. However, except for their ends, carbon nanotubes are chemically inert. To make chemically selective probes there is interest in studying substitution reactions for nanotube tips. If that chemistry can be controlled, scientists should be able to design highly selective single-molecule probes.

The goal of this project was to determine the properties of carbon nanotubes with chemically modified ends. Using quantum chemistry techniques, the calculations were carried out for fragments of nanotubes with amide and ester groups attached.

For each functional group, the aim was to determine the lowest optimized energy for each functional group, with respect to the corresponding nanotube size. For each size (16-0, 10-0, 10-10), the functional groups were at different torsional angles (e.g. 90, -90/ -90,90) to obtain the significant measurements for concise analysis. Next, the energies were plotted with respect to the different torsions.

In conclusion, it clearly seen how carbon nanotubes contain a range of properties that make them well suited for use as probe tips in AFM applications. The preparation of a wide range of functionalized nanotube tips could provide the model for molecular probes with usage in many areas of chemistry and biology. Using calculation methods as previously mentioned, scientists can modify nanotubes to create probes that can manipulate matter at the molecular level. Future studies could possibly determine bond strengths for functional groups and derive force fields for molecular dynamics calculations of functionalized nanotubes in aqueous solutions. Furthermore, carbon nanotubes offer amazing possibilities to create future nanoelectronics devices, computers, sensors and the recognition platforms for detecting biomolecules.

4 Metabolic Profiling by Multidimensional NMR

(point of contact: Shoudan Liang)

Vivek Guruswamy, U.C. Berkeley

4.1 Introduction

Although the knowledge of genes and their similiar statement patterns is useful information, it would be much more useful if there were a way to find a correlation between that information and the different metabolic processes. If this could be done, then much more light could be shed on the unknown functions that genes or groups of genes play in life. The basis of this project is to realize the different roles genes play in the regulation of metabolic processes. In order to do this, it has been proposed that different computational techniques and multidimensional NMR be developed in order to help locate different metabolites. The metabolites would help, since they are a direct characterization of its metabolic functions. The data obtained from the NMR could be used with the DNA micro-array gene statement data to help interpret the metabolites.

Developing algorithms which calculate the peaks in a two-dimensional NMR spectrum is a promising new step in metabolic profiling. The algorithms will be developed so that they identify a group of peaks that are associated with a certain compound. In order to help in creating these algorithms, 16 2-Dimensional NMR spectra have been developed by Professor Peng (University of Connecticut Health Center). These spectra each contain a random mixture of 20 naturally occurring amino acids. Using the algorithms which will be developed, the peaks in each spectrum must be analyzed and grouped together to see which peaks belong to the same amino acid and how much of each amino acid are in each spectrum.

4.2 Developing the Algorithm

A. Reading the Data. The first step in creating the algorithm required reading the binary data given by Professor Peng. To make sure the data was read in correctly, it was plotted and compared with the plots given by Professor Peng. To read the data, a simple program (in C++) was created. This program first read in the 512 bit header file to pass it over to get the raw data. In this code, header was defined as a 512 x 1 array.

Once the header had been read in, the real data from the NMR spectrum was read into a 1024 x 4096 matrix. Each piece of data (each number) was depicted as a 4 byte character string in order to properly read it in as binary data. The string was then converted to a floating point and read into a matrix.

After this process was finished, the data had to be checked to see if it was read in properly. It was checked by comparing the plotted data from the matrix, and the plots given by Professor Peng.

The plotting was achieved by using MATLAB. The data from the matrix was first printed out to a file. This file was read in using MATLAB and then plotted as a contour plot. The plot created matched the plot given, which proved that the data had been read in correctly. These plots can be seen in the attached figures (in contour plot and mesh plot).

B. Locating the Peaks. After creating the matrix, the next step was to locate the peaks and find the position and height of the peak. To do this the matrix had to be run through a loop to find the maximum values, which would be the peaks. Finding the maximum values required a recursive function which went through the matrix and found maximum values. Once the maximum value was found, a variable was created which equaled the 10(it is only until 10% because everything below that is assumed to be noise) was "tagged" so that it wouldn't be counted when the next peak was being searched for.

The function takes in the x y values for the maximum value, and the maximum value itself. When the function is called, the recursive code looks in the four directions (up, down, left, right) to see if that value is under and part of the peak. Although this method has not been perfected, it gives a good idea of where the peaks are located. The function also calculates the value of the width and length of the base of the peak. These values are used in the next step. The loop searches for peaks until the defined number of peaks has been reached.

C. Using the Conjugate Gradient to fit the Peaks Once the data (peak height and coordinates of peaks) have been collected, they are used as input for the conjugate gradient

algorithm. The code for the conjugate gradient algorithm was used from "Numerical Recipes for C". It was modified to fit with the parameters of the NMR data. Although it was not fully implemented, due to conflicts in modifying the program from C to C++, a general idea of how the algorithm should be written was achieved.

The conjugate algorithm was used in order to optimize the parameters estimated from the spectrum data. The parameters include the dimensions of the base of the peak, the height of the peak, and the coordinates of the peak. In order to fit the peaks into the spectrum, a cost function is needed. This cost function is equal to the gaussian function of the peak subtracted from the intensity of the peak squared. The derivatives of the gaussian function with respect to all the parameters were also calculated in order to use the conjugate gradient algorithm.

4.3 Conclusion

Due to various conflicts (code modification, translation from C to C++), the entire program could not be completed. However what was achieved can be used towards developing a good algorithm which is capable of locating and calculating 2D spectrum peaks. The analysis of the spectrum peaks is a new step in metabolic profiling which can lead to the discovery of unknown gene functions.

4.4 References

Press WH, Teukolsky SA, Vetterlink WT, Flannery BP. *Numerical Recipes in C: The Art of Scientific Computing* (Cambridge University Press, Cambridge, 1992) pp. 4200-424.

Schildt, Herbert. *C++: The Complete Reference, Third Edition.* (McGraw-Hill Professional Publishing, New York, 1998).

5 Structural Homology Prediction and MD Modeling to Find GdETmod Structure

(point of contact: Michael New or Andrew Pohorille)

Ryan J. Weber, U.C. Santa Cruz

5.1 Introduction

Here we are trying to find the structure of the Tropomodulin gene family and since "all Tmod isoforms are of nearly the same length and display very few gaps and insertions relative to one another,"(Conley) we can focus on *E-Tmod.* This sub-category of Tropomodulin is primarily expressed in striated muscle. In particular, the sequence used here will be denoted *GdETmod.* However, an alignment of eight [1] similar Tmods, Lmods, and Cmods is used to

[1] Actually nine sequences are used for modeling the NH-terminus including CionaTmod

locate more remote homologs that can be used as initial structural models for GdETmod. These eight sequences are DmTmod, DrSkTmod, GdETmod, HmSMLmod, MmCLmod, RnNTmod, SsUTmod, and embCel. The full alignment of these sequences is given here as created by SAMs *prettyalign*:

```
HsSMLmod ----------------------MSRVAKYRRQVS---EDPDIDSLLETLSPEEMEELEKEL
MmCLmod  ----------------------MSTFGYRRGLSK-YESIDEDELLASLSPEELKELEREL
GdETmod  -------------------------MSYRKELEK-YRDLDEDKILGALTEEELRKLENEL
RnNTmod  ------------------------MALPFQKGLEK-YKNIDEDELLGKLSEEELKQLENVL
SsUTmod  ------------------------MALSFRKDLEK-YKDLDEDELLGNLSEVELKQLETVL
DrSkTmod -------------------------MSKSDP-----RDIDEDAILRGLSAEELEQLDIEL
DmTmod   ----------METSATTKTTTLTTPAKLYGKDLS-EYDDVDVESLLAQLSPE---EITILA
embCe1   MSQAKTDYYSEEKTFSAPSANSQQGTQLPSKVYNKGLKDLEDNDIEGLLSSLSIDELEDLN


                70        80        90       100       110       120
                 |         |         |         |         |         |
HsSMLmod DVVDPDG-SVPVGLRQRNQTEKQSTGVYNREAMLNFCEKETKKLMQREMSMDESKQVETKT
MmCLmod  EDIEPDR-NLPVGLRQKSLTEKTPTGNFSREALMAYWEKESQKLLEKERLGECGK------
GdETmod  EELDPDNALLPAGLRQRDQTQKPPTGPFKREELMAHLEQQAKDIKDREDLVPFT-------
RnNTmod  DDLDPESATLPAGFRQKDQTQKAATGPFDREHLLMYLEKEALEQKDREDFVPFT-------
SsUTmod  DDLDPENALLPAGFRQKNQTSKSATGPFDREHLLSYLEKEALEHKDREDYVPYT-------
DrSkTmod QELDPENTTLPAGFRQRDQTKKSPTGPFDRFALMDYLEKQAIEHKDRDDLVPFT-------
DmTmod   KEVDPDDNFLPPDQRNSYECTKEATGPLNRKQLIEHINKQAIETPDQPEFEPFVQ------
embCe1   NDFDPDNSMLPPSQRCRDQTDKEPTGPYKRDNLLKFLEDKAKTEKDWEDVCPYTP------


               130       140       150       160       170       180
                 |         |         |         |         |         |
HsSMLmod DAKNGQERGRDASKKALGPRRNSDLGKEPKRGGLKKSFSRDRDEAGGKSGEKPKEEKIIRG
MmCLmod  ----VAEEDKEESEE-------------------ELIFTESNSEVSEEVCTEDEEE-----
GdETmod  ----GEKRGKAWIP--------------------KQKPMDPVLE---SVTLEP--------
RnNTmod  ----GEKKGRVFIP--------------------KEKPVETRKEE--KFTLDP--------
SsUTmod  ----GEKKGKIFIP--------------------KQKPVQTFTEE--KVSLDP--------
DrSkTmod ----GEKRGKAFVP--------------------KPGSGQIPADE---QITLEP--------
DmTmod   ----GKVRGKKWVPP-----------------PRDARDIEAEEQI----AIDMG-------
embCe1   ----GQKRGKVYDSD---------------SGRNSEEPENGKMEMPIEIDLDDDEE-----


              190       200       210       220       230       240
                 |         |         |         |         |         |
HsSMLmod IDKGRVRAAVDKKEAGKDGRGEERAVATKKEEEKKGGDRNTGLSRDKDKKREEMKEVAKKE
MmCLmod  ------------------SQEEEEDSEEEEDSEEE------------EETTEATKHINGTV
GdETmod  --------------------ELEEALANASDAE--------------------LCDIAAIL
RnNTmod  --------------------ELEEALASASDTE--------------------LYDLAAVL
SsUTmod  --------------------ELEEALTSASDTE--------------------LCDLAAIL
```

```
DrSkTmod --------------------ELEEALRNATDAE--------------------MCDIAAIL
DmTmod   --------------------EEYEHALNDATQEE--------------------IIDLAAIL
embCe1   --------------------ELECALVTAPEKD--------------------LVDLAGIL


            250       260       270       280       290       300
              |         |         |         |         |         |
HsSMLmod DDEKVKGERRNTDTRKEGEKMKRAGGNTDMKKEDEKVKRGTGNTDTKKDDEKVKKNEPLHE
MmCLmod  SYNSVNTD----------------------NSKPKTFKSQIENINLTNGNSGRTQRN----
GdETmod  GMHTLMSN----------------------QQYYEALGSSTIV--NKEGLNSVIKPT----
RnNTmod  GVHNLLNN----------------------PKFDEETTNGQG---RKGPVRNVVKGE----
SsUTmod  GMHNLITN----------------------TQFCNIVGSSNGV--DQEHFSNVVKGE----
DrSkTmod GMYTLMSN----------------------KQYYDALNTTGKI-ANTEGINSVVKPD----
DmTmod   GFHSMMNQ----------------------DQYHASLLNKGQP--VGLGWDGITKST----
embCe1   GMHNVLNQ----------------------PQYYNALKGKTQDESTGTTFNGIMQSY----


            310       320       330       340       350       360
              |         |         |         |         |         |
HsSMLmod KEAKDDSKTKTPEKQTPSGPTKPSEGPAKVEEEAAPSIFDEPLERVKNNDPEMTEVNVNNS
MmCLmod  ------------------------SESPAAIHPCGNPTVIEDALEKIRNNDPDTTEVNLNNI
GdETmod  ------------------------KYKPVPDEEP-NSTDVEETLKRIQNNDPDLEEVNLNNI
RnNTmod  ------------------------KAKPVFEEPP-NPTNVEASLQQMKANDPSLQEVNLNNI
SsUTmod  ------------------------KILPILDEPP-NPTNVEESLKRIKEDDVRLVEVNLNNI
DrSkTmod ------------------------VYKIYPEEPP-NDTNVEETLRYIQKNDNRLQEVNLNNI
DmTmod   ------------------------QQKLFPMDPP-NNTDVEESIKRVKDDDSKLIDLNLNNI
embCe1   ------------------------VPRIVPDEPD-NDTDVESCINRLREDDTDLKEVNINNM


            370       380       390       400       410       420
              |         |         |         |         |         |
HsSMLmod DCITNEILVRFTEALEFNTVVKLFALANTRADDHVAFAIAIMLKANKTITSLNLDSNHITG
MmCLmod  ENITTQTLSRFAEALKENTVVKTFSLANTHADDAAAIAIADMLKVNEHITSVNVESNFITG
GdETmod  MNIPVPTLKACAEALKTNTYVKKFSIVGTRSNDPVAFALAEMLKVNNTLKSLNVESNFISG
RnNTmod  KNIPIPTLKEFAKALETNTHVRKFSLAATRSNDPVALAFAEMLKVNKTLKSLNVESNFITG
SsUTmod  KNIPIPTLKDFAKALETNTHVKYFSLAATRSNDPVAAAFADMLRVNKNLKSLNMESNFITG
DrSkTmod PDIPIPTLKEIFEAMKRNTHVLCLSIAGTRSNDPVAYAIAEMLQANKNLQSLNIESNFITA
DmTmod   KNISDEKLEQLFAALPQNEHLEVLSLTNVGLTDKTALLLAAAIEKSKTLRVLNVETNFISP
embCe1   KRVSKERIRSLIEAACNSKHIEKFSLANTAISDSEARGLIELIETSPSLRVLNVESNFLTP


           430       440       450       460       470       480       4
              |         |         |         |         |         |
HsSMLmod KGILAIFRALLQNNTLTELRFHNQRH-ICGGKTEMEIAKLLKENTSLLKLGYHFELAGPRM
MmCLmod  KGILAIMRALQHNTVLTELRFHNQRH-IMGSQVEMEIVKLLKENTTLLRLGYHFELPGPRM
GdETmod  SGILALVEALQSNTSLIELRIDNQSQ-PLGNNVEMEIANMLEKNTTLLKFGYHFTQQGPRL
```

```
RnNTmod  AGILALVEALRENDTLTEIKIDNQRQ-QLGTAVEMEIAQMLEENSRILKFGYQFTKQGPRT
SsUTmod  AGILALMDALRDNETLAELKIDNQRQ-QLGTAVELEMAKMLEENTNILKFGYQFTQQGPRT
DrSkTmod QGMMAIVKALRKNSTLIEIKIDNQRQ-KLGDSVEMEIASMLEKNSSIIKIGYHFTQQGPRA
DmTmod   PVIVKLVQALLKCHTIEEFRASNQRSAVLGNKIEMEITDLVEKNSSLLRLGLHLEFNDARH
embCe1   ELLARLLRSTLVTQSIVEFKADNQRQSVLGNQVEMDMMMAIEENESLLRVGISFASMEARH

          90       500       510       520       530       540       55
           |         |         |         |         |         |
HsSMLmod TVTNLLSRNMDKQRQKRLQEQRQAQEAKGEKKDLLEVPKAGAVAKGSPKPSPQPSPKPSPK
MmCLmod  SMTSILTRNMDKQRQKRMQEQKQQEGHDGGAALRTKVWQRGTPG-SSPYASPRQSPWSSPK
GdETmod  RASNAMMNNNDLVRKRRLAELNGPIFPKCRTGV----------------------------
RnNTmod  RVAAAITKNNDLVRKKRVEGDRR-------------------------------------
SsUTmod  RAANAITKNNDLVRKRRVEGDHQ-------------------------------------
DrSkTmod RAAMAITRNNVILRQQRVR-----------------------------------------
DmTmod   RVAAHLQRNIDRIRVKRLNQRK-------------------------------------
embCe1   RVSEALERNYERVRLRRLGKDPNV-------------------------------------

          0       560       570       580       590       600       610
          |         |         |         |         |         |         |
HsSMLmod NSPK------KGGAPAAPPPPPPP-------------------------------------
MmCLmod  VSKKVHTGRSRPPSPVAPPPPPPPPPPLPPHMLPPPPPPPAPPLPEKKLITRNIAEVIKQQE
GdETmod  ------------------------------------------------------------
RnNTmod  ------------------------------------------------------------
SsUTmod  ------------------------------------------------------------
DrSkTmod ------------------------------------------------------------
DmTmod   ------------------------------------------------------------
embCe1   ------------------------------------------------------------

                 620       630       640       650       660       670
                   |         |         |         |         |         |
HsSMLmod ------------------LAPPLIMENLKNSLSPATQRKMGDKVLPAQEKNSRDQ-LLAA
MmCLmod  SAQRALQNGQRKKKGKKVKKQPNNILKEIKNSLRSVQEKKMEDSSRPSTPQRSVHENLMEA
GdETmod  ------------------------------------------------------------
RnNTmod  ------------------------------------------------------------
SsUTmod  ------------------------------------------------------------
DrSkTmod ------------------------------------------------------------
DmTmod   ------------------------------------------------------------
embCe1   ------------------------------------------------------------

               680       690
                 |         |
HsSMLmod IRSSNLKQLKKVEVPKLLQ
```

```
MmCLmod  IRGSSIRQLRRVEVPEALR
GdETmod  -------------------
RnNTmod  -------------------
SsUTmod  -------------------
DrSkTmod -------------------
DmTmod   -------------------
embCe1   -------------------
```

5.2 Finding Related Structures

The exact amino acid sequence of length 79 is given here for a region of the carboxy-terminus of GdETmod:

```
DEEPNSTDVEETLERIKNNDPKLEEVNLNNIRNIPIPTLKAYAEALKENS
YVKKFSIVGTRSNDPVAYALAEMLKENKVLKTLNVESNFISGAGILRLVE
ALPYNTSLVEMKIDNQSQPLGNKVEMEIVSMLEKNATLLKFGYHFTQQGP
RLRA
```

First it is important to find other sequences with a known structure that are similar to this one. A sequence can be found based on its amino acid similarity with substitution matrices such as PAM or BLOSUM, or according to pre-defined structural motifs that are often expressed as regular expressions.

5.2.1 Cassie's Original Ranking: UCLA/DOE Fold Server

The UCLA/DOE fold server at

`http://fold.doe-mbi.ucla.edu/`

is one method for finding likely structural matches for GdETmod with unknown structure. It returns a list of 500 top for each member of the hand-tuned alignment of eight different Tmods, Cmods, and Lmods. The original algorithm used by the UCLA/DOE fold server would look for small motif matches and it would return E-values and numerical rankings for these top 500 hits.

Cassie made the first run and compiled the summation of all the top 20 rankings into a table, giving the value 21 to those not in the top 20 of all eight query sequences. This clearly showed that 2bnh was the closest match on all structures except *Spdo*. In addition, 1gky also scores quite highly. However, these two choices are based on ranking not Z-score. Figure 1 shows 1gky and Figure 2 shows 2bnh.

5.2.2 New and Questionably Improved UCLA/DOE Fold Server

Recent changes to the folder server, which are beyond our control, cause it to only look for a global match and it is therefore not as sensitive to smaller motif matches. The top 500 matches that the folder server returns for each of the query sequences are summed by rank,

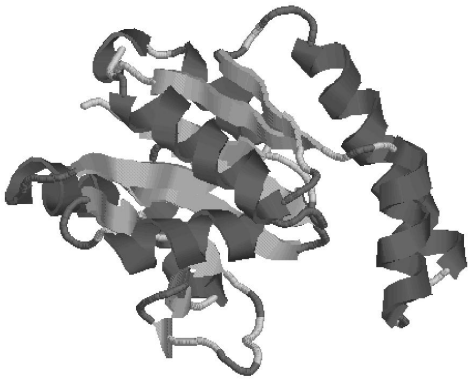

Figure 1: 1GKY

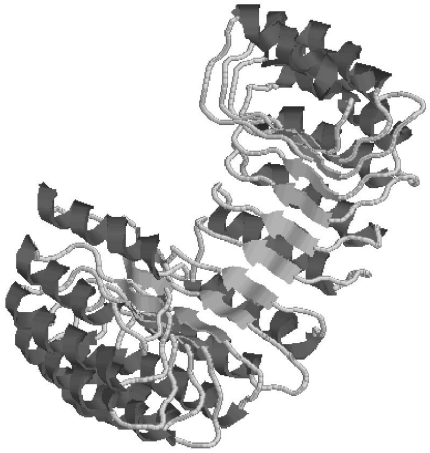

Figure 2: 2BNH

as Cassie did to get 2bnh and 1gky. Here are the top 5 scoring proteins with known structure and the scores each of the eight query sequences get on them. The Z-scores are shown in parenthesis and the rankings are in square brackets. These top 5 are chosen based on the summation of Z-scores:

```
2ts101: 8 (0.023894) [170]
------
DmTmod.500 (0.00373351) [1]
DrSkTmod.500 (0.00282138) [14]
GdETmod.500 (0.0025288) [52]
HsSMLmod.500 (0.00307739) [3]
MmCLmod.500 (0.00322639) [5]
RnNTmod.500 (0.00285677) [11]
SsUTmod.500 (0.003277) [3]
```

```
embCe1.500 (0.00237277) [81]

2chr02: 8 (0.0232777) [130]
------
DmTmod.500 (0.00301463) [6]
DrSkTmod.500 (0.00296071) [6]
GdETmod.500 (0.00290694) [14]
HsSMLmod.500 (0.00242952) [53]
MmCLmod.500 (0.00275897) [28]
RnNTmod.500 (0.0027975) [15]
SsUTmod.500 (0.00318337) [5]
embCe1.500 (0.00322603) [3]

1ad1B0: 8 (0.0228929) [228]
------
DmTmod.500 (0.00335088) [2]
DrSkTmod.500 (0.00258917) [29]
GdETmod.500 (0.00241064) [75]
HsSMLmod.500 (0.00290385) [7]
MmCLmod.500 (0.00235994) [96]
RnNTmod.500 (0.00289233) [10]
SsUTmod.500 (0.00325359) [4]
embCe1.500 (0.00313252) [5]

1rpxA0: 8 (0.022691) [160]
------
DmTmod.500 (0.00262041) [28]
DrSkTmod.500 (0.00263561) [26]
GdETmod.500 (0.00300148) [9]
HsSMLmod.500 (0.00291542) [6]
MmCLmod.500 (0.0032492) [3]
RnNTmod.500 (0.00259599) [36]
SsUTmod.500 (0.00307804) [7]
embCe1.500 (0.00259485) [45]

1ctqA0: 8 (0.0226376) [196]
------
DmTmod.500 (0.00245809) [53]
DrSkTmod.500 (0.00256595) [32]
GdETmod.500 (0.00335598) [2]
HsSMLmod.500 (0.00255678) [29]
MmCLmod.500 (0.00318079) [8]
RnNTmod.500 (0.00301087) [6]
```

```
SsUTmod.500 (0.00297271) [11]
embCe1.500 (0.00253641) [55]
```

To summarize, the top 5 matching sequences with known structure from Pdb are: 2ts101, 2chr02, 1ad1B0, 1rpxA0, 1ctqA0. These structures are illustrated in Figure 3. None of the structures clearly scores best on all of the Tmods, Cmods, and Lmods. This is also illustrated in Figure 4. It shows the $-log(Evalues)$ for each of the 500 hits, with respect to each of the eight novel sequences. Picking the best matches based on rank, instead of Z-score, produces similar results.

Strangely, the structures 2bnh and 1gky do not appear in any of the top 500 lists with these new runs on the UCLA/DOE fold server. Cassie's confidence in the original scoring (using the smaller local matches and a cutoff of 21 for the top 20 hits) allows us to continue by comparing the behavior of GdETmod when threaded onto 2bnh and 1gky.

5.3 Building the Model

Using Modeller we create the models that will be used in minimization and MD for the carboxy-terminus of GdETmod. This process requires an alignment between the GdETmod sequence and the model sequence (either 1gky or 2bnh). Modeller includes a tool for automatically creating an alignment but here we rely on hand-crafted alignments guided by the conservation of predicted secondary structure elements.

The input file given to Modeller for 2bnh also shows the alignment:

```
>P1;eTmod
sequence:eTmod
DEEPNSTDVEETLERIKNNDPKLEEVNLNNIRNIPIPTLKAYAEALKENS
YVKKFSIVGTRSNDPVAYALAEML-KENKVLKTLNVESNFISGAGILRLV
EALPYNTSLVEMKIDNQSQPLGNKVEMEIVSMLEKNATLLKF----GYHF
TQQGPRLRA*

>P1;2bnh
structureX:2bnh:232: :387: :
SNGLGDAGIAELCPGLLSPASRLKTLWLWECD-ITASGCRDLCRVLQAKE
TLKELSLAGNKLGDEGARLLCESLLQPGCQLESLWVKSCSLTAACCQHVS
LMLTQNKHLLELQL--SSNKLGDSGIQELCQALSQPGTTLRVLCLGDCEV
TNSGCSSLA*
```

The numbers after 2bnh refer the the starting and ending amino acid positions in the pdb file for 2bnh. Similarly, the input file for 1gky is:

```
>P1;eTmod
sequence:eTmod
DEEPNSTDVEETLERIKNNDPKLEEVNLNNIRNIPIP----------TL
```

```
KAYAEALKENSYVKKFSIVGTRSNDPVAYALAEMLKENKVLKTLNVESNF
ISGAGILRLVEALPYNTSLVEMKIDNQSQPLGNKVE---MEIVSMLEKNA
TLLKFGYHFTQQGPRLRA*

>P1;1gky
structureX:1gky:7: :166: :
ISGPSGTGKSTLLKKLFAEYPDSFGFSVSSTTRTPRAGEVNGKDYNFVSV
DEFKSMIKNNEFIEWAQFSGNYYGSTVASVKQVSKSGKTCILDIDMQG--
------VKSVKAIPELNARFLFIAPPSVEDLKKRLEGRGTETEESINKRL
SAAQAELAYAETGAHDKV*
```

Notice the relatively large gap in the eTmod when aligned to the 1gky and another smaller gap in the 1gky sequence itself. Only a few smaller gaps exist with the 2bnh alignment shown above. This provides some confidence that the 2bnh is actually a closer structure, but it is also an artifact of the alignment algorithm and parameters.

The actual Modeller script is very simple:

```
INCLUDE
SET ALNFILE = 'other.ali'
SET KNOWNS  = '1gky'
SET SEQUENCE = 'eTmod'
SET ATOM_FILES_DIRECTORY = './'
SET STARTING_MODEL = 1
SET ENDING_MODEL = 1
CALL ROUTINE = 'model'
```

for alignment file "other.ali" and with '1gky' replaced by '2bnh' for that protein's model of eTmod.

5.4 Protonation and Equalizing the Charge

Before minimizing the new models it is necessary to add protons using the command

```
protonate -d $AMHOME/dat/PROTON_INFO < prot.pdb > prot.H.pdb
```

Given the input file "prot.pdb", representing one of the two eTmod threaded models, `protonate` outputs "prot.H.pdb". The environment variable $AMHOME must be defined as the Amber home directory.

The program xLeap can add sodium (Na+) ions to the variable representing the input file "prot.H.pdb" called *prot* with the command:

```
addIons prot Na+ 3
```

In this case three are needed to balance out the charge for both the model based on 2bnh and the model based on 1gky. The final structure of *prot* is then converted into a topology file (prmtop) and a coordinate file (prmcrd) using xLeap. The initial threaded structures are shown in Figure 5 and Figure 6 in cartoon format with the colors corresponding to their secondary structures. These are used as input to Sander for minimization.

5.5 Minimization

Thirty iterations of 2500 step minimization are performed using the Sander input script shown here:

```
# Minimization Script (title is necessary)
  &cntrl
     imin=1,          //do minimization
     nmrmax=0,        //don't do nmr
     ntpr=20,         //print energy info every 20 steps
     ntb=0,           //no periodicity
     idiel=0,         //distance dependent dialectic
     dielc=80.0,      //dielectric value
     cut=12.0,        //non-bonded cutoff dist. (d)
     ntnb=1,          //write non-bonded pair list (d)
     nsnb=25,         //write pair-list every 25ns (d)
     scnb=2.0,        //vdw divisor (d)
     scee=1.2,        //electrostatic interaction divisor (1994 d)
     maxcyc=2500,     //2500 cycles of minimization
     ntmin=0,         //full conjugate gradient minimization
     ncyc=2500,       //Only steepest descent
     dx0=0.01,        //initial step length(d)
     dxm=0.5,         //maximum allowed step length (d)
     drms=.1,         //stopping criteria
     ntc=1,           //Don't use SHAKE (d)
  &end
```

The comments must be removed from the script before running it and the (d) in the comments refers to a default option that should be set that way initially if not explicitly set in the input script. For a more detailed description of the scripting commands see the Modeller4 manual.

Output from the minimization runs is shown for the 1gky model in Figure 7 and the 2bnh model in Figure 8. One key point is that the 1gky model was minimized with all bonds involving hydrogen constrained and SHAKE on. This is not the best thing to do, but it most likely has little effect in this case.

5.6 Molecular Dynamics

The MD process can be divided into two phases. The first heats the protein up and the the second phase equilibrates the heated structure. Heating too fast can make the entire structure simply fall apart, forming a glob.

5.6.1 Heating

The heating process is divided into six iterations, where the total temperature change is from 0K to 300K. Each iteration changes the temperature 50K over a period of 10 picoseconds and rescaling is performed in between iterations. The script used for the first iteration of this process is:

```
# MD Heating:  10.0 ps
 &cntrl
  imin=0, nrun = 10,
  ntx=1, irest=0, ntrx=1,
  ntxo=1, ntpr=25, ntwr=1000, ntwx=100, ntwe=100,
  ntf=2, ntb=0, dielc=80.0, idiel=0, scnb=2, scee=1.2,
  nstlim=1000, ntcm=1, nscm=0, t=0.0, dt=0.001,
  temp0=50.0, tempi=0.0, ntt=0, dtemp=5.0,
  ntp=0, taup=0.2,
  ntc=2, tol=0.0005,
  cut=12.0, nsnb=25,
 &end
```

One small difference in subsequent iterations is the value of "ntx" is changed from 1 to 5 and the value of "irest" is changed from 0 to 1. This is small change based only on the input format expected when restarting a heating run as opposed to using the final output of minimization. It has no effect on the underlying MD algorithm.

Output from the heating runs is shown in Figure 10 and Figure 11 for the 1gky model and the 2bnh model respectively. Already, the model based on the 1gky has lost a lot of its structure and the 2bnh model has lost some of the beta sheets in the center. The total energy and temperature throughout the process is shown for the 1gky model in Figure 9.

5.6.2 Equilibration

After heating, equilibration is necessary hopefully find a stable global minimum energy structure (or show that none exists). This process is performed in two iterations with the same Sander script:

```
# MD Equilibration: 100.0 ps
 &cntrl
  imin=0, nrun=10,
  ntx=5, irest=1,
  ntpr=100, ntwr=100, ntwx=100,
  nsnb=25, cut=12.0, cut2nd=14.0, scee=1.2,
  ntb=0,
  nstlim=5000,
  ntt=1, tempi=297.0, temp0=297.0, tautp=0.2,
```

```
  ntf=2, ntc=2, dtemp=5.0,
  dt=0.002, ntwxm = 0,
 &end
```

Unfortunately none of these simulations were sucessful. The scripts were tested on what turned out to be the wrong structure and they worked, but not with the output of these particular heating runs. This may be due to the escaping sodium ions that make the output pdb files impossible to view without explicity removing them. Perhaps another method of equalizing the charge is more appropriate. Therefore there was not possible to create pdb output for these runs.

5.7 Amino-Terminus

Now as a side note, we attempt to find the structure for the Amino-terminus half of all the eight or nine query sequences. Contact Cassie Conley for the full-alignment. Here are the top 5 scoring proteins with known structure and the scores each of the eight query sequences get on them. The Z-scores are shown in parenthesis and the rankings are in square brackets.

```
1qhfA0: 8 (0.0228278) [401]
------
DmTmod.500 (0.00328202) [2]
DrSkTmod.500 (0.00307936) [4]
GdETmod.500 (0.00244043) [86]
HmSMLmod.500 (0.00271905) [51]
MmCLmod.500 (0.00396204) [2]
RnNTmod.500 (0.00276913) [21]
SsUTmod.500 (0.00214828) [149]
embCe1.500 (0.00242748) [86]

1frb00: 8 (0.0210617) [479]
------
DmTmod.500 (0.00230092) [87]
DrSkTmod.500 (0.0025925) [37]
GdETmod.500 (0.00227617) [125]
HmSMLmod.500 (0.00258113) [66]
MmCLmod.500 (0.00362146) [7]
RnNTmod.500 (0.00237861) [68]
SsUTmod.500 (0.00261374) [50]
embCe1.500 (0.00269721) [39]

3pgm00: 8 (0.0215033) [533]
------
DmTmod.500 (0.00287323) [8]
DrSkTmod.500 (0.00294547) [9]
```

```
GdETmod.500 (0.00246389) [78]
HmSMLmod.500 (0.00237424) [117]
MmCLmod.500 (0.00362146) [6]
RnNTmod.500 (0.00220111) [130]
SsUTmod.500 (0.00212441) [160]
embCe1.500 (0.0028995) [25]

1c9wA0: 8 (0.0201347) [628]
------
DmTmod.500 (0.00245276) [59]
DrSkTmod.500 (0.00236125) [87]
GdETmod.500 (0.00226443) [128]
HmSMLmod.500 (0.00236439) [125]
MmCLmod.500 (0.003349) [10]
RnNTmod.500 (0.00242595) [60]
SsUTmod.500 (0.0026018) [52]
embCe1.500 (0.0023151) [107]

1zin00: 8 (0.0207094) [647]
------
DmTmod.500 (0.00230092) [85]
DrSkTmod.500 (0.00278724) [22]
GdETmod.500 (0.00247563) [76]
HmSMLmod.500 (0.00192107) [244]
MmCLmod.500 (0.00217969) [141]
RnNTmod.500 (0.00314782) [2]
SsUTmod.500 (0.00341338) [5]
embCe1.500 (0.00248368) [72]
```

And here are the top 5 summed results using the normalized Z-scores instead of the rank:

```
1qhfA0: 8 (0.0228278) [401]
------
DmTmod.500 (0.00328202) [2]
DrSkTmod.500 (0.00307936) [4]
GdETmod.500 (0.00244043) [86]
HmSMLmod.500 (0.00271905) [51]
MmCLmod.500 (0.00396204) [2]
RnNTmod.500 (0.00276913) [21]
SsUTmod.500 (0.00214828) [149]
embCe1.500 (0.00242748) [86]

3pgm00: 8 (0.0215033) [533]
------
```

```
DmTmod.500 (0.00287323) [8]
DrSkTmod.500 (0.00294547) [9]
GdETmod.500 (0.00246389) [78]
HmSMLmod.500 (0.00237424) [117]
MmCLmod.500 (0.00362146) [6]
RnNTmod.500 (0.00220111) [130]
SsUTmod.500 (0.00212441) [160]
embCe1.500 (0.0028995) [25]

1frb00: 8 (0.0210617) [479]
------
DmTmod.500 (0.00230092) [87]
DrSkTmod.500 (0.0025925) [37]
GdETmod.500 (0.00227617) [125]
HmSMLmod.500 (0.00258113) [66]
MmCLmod.500 (0.00362146) [7]
RnNTmod.500 (0.00237861) [68]
SsUTmod.500 (0.00261374) [50]
embCe1.500 (0.00269721) [39]

1zin00: 8 (0.0207094) [647]
------
DmTmod.500 (0.00230092) [85]
DrSkTmod.500 (0.00278724) [22]
GdETmod.500 (0.00247563) [76]
HmSMLmod.500 (0.00192107) [244]
MmCLmod.500 (0.00217969) [141]
RnNTmod.500 (0.00314782) [2]
SsUTmod.500 (0.00341338) [5]
embCe1.500 (0.00248368) [72]

1vom05: 8 (0.0203922) [1012]
------
DmTmod.500 (0.00198556) [198]
DrSkTmod.500 (0.00265336) [33]
GdETmod.500 (0.0029684) [10]
HmSMLmod.500 (0.00147775) [414]
MmCLmod.500 (0.0031333) [21]
RnNTmod.500 (0.00299398) [7]
SsUTmod.500 (0.00344918) [4]
embCe1.500 (0.00173071) [325]
```

So to summarize, choosing the minimum sum of ranks from the set of proteins that are in the intersection of all eight top 500 lists gives top 5: 1qhfA0, 1frb00, 3pgm00, 1c9wA0, and 1zin00, in that order. Choosing the maximum sum of normalized Z-scores from all the proteins in the union of the eight top 500 lists gives top 5: 1qhfA0, 3pgm00, 1frb00, 1zin00, and 1vom05 in that order. The top hit, 1qhfA0, is still best for both scoring methods and the second and third hits are identical but in reverse order. However, there doesn't appear to be an obvious connection between these top scoring structures. Graphs of the $-log(Evalues)$ for each of the eight sequences and CionaTmod are shown in Figure 12 with a line drawn horizontally, dividing the top 10 hits from the others. The top two structures according to Z-score (1qhf and 3pgm) are given in Figure 13 and Figure 14.

5.7.1 Pfam-Vector

As a side experiment, we scored a few of the novel sequences against 2866 HMMs built from the Pfam-A seeds. The $-log(Evalues)$ are compared in Figure 15 for HsSMLmod, MmCLmod, and embCel. The top 5 Pfam IDs for each of the queries are shown here:

```
HsSMLmod: PF02674, PF01547, PF00854, PF00915, PF00251
MmCLmod : PF00118, PF00533, PF02665, PF01486, PF01047
embCel  : PF02009, PF00817, PF02489, PF01068, PF01476
```

Unfortunately there is no clear line between the best hits and the background noise, and the intersection between the three sequences top 5 is empty. However, looking further into these Pfam families may uncover some connection.

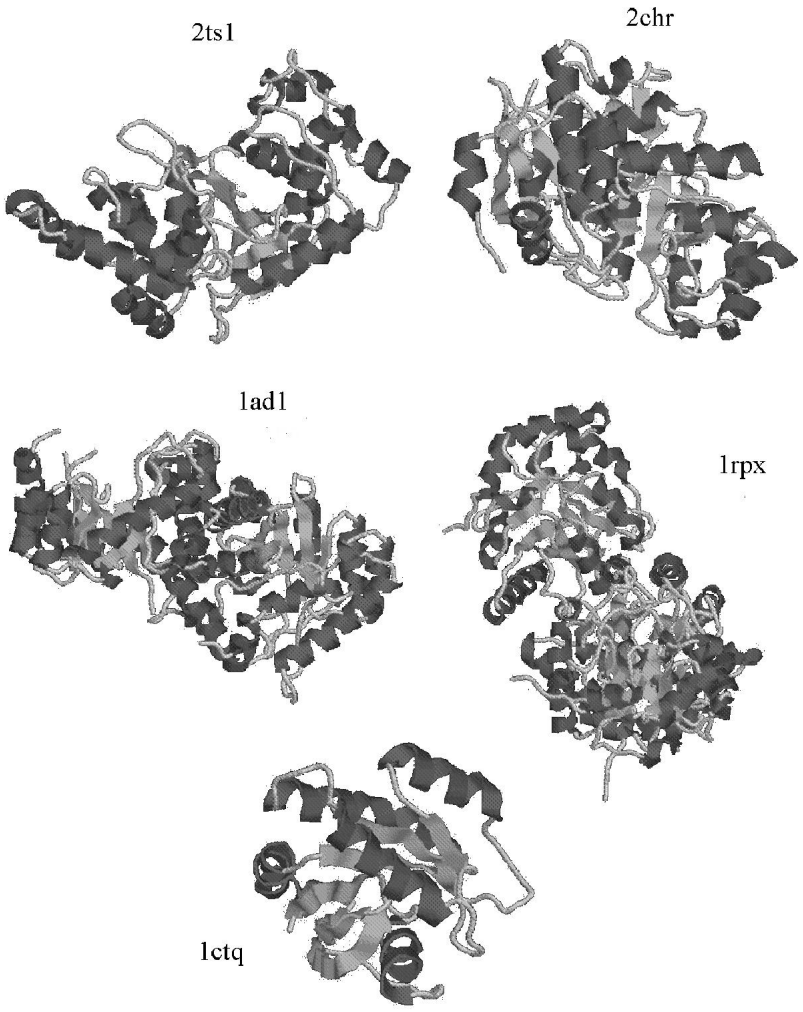

Figure 3: These five structures are the highest scoring, according to the maximum sum of the Z-scores from each of the eight query sequences, for the carboxy-terminus of GdETmod. They all share some beta strands in the center with alpha helices around them, but they are still distinctly different. PDB is only concerned with the first 4 letter of each identifier so the last 2 characters are ignored.

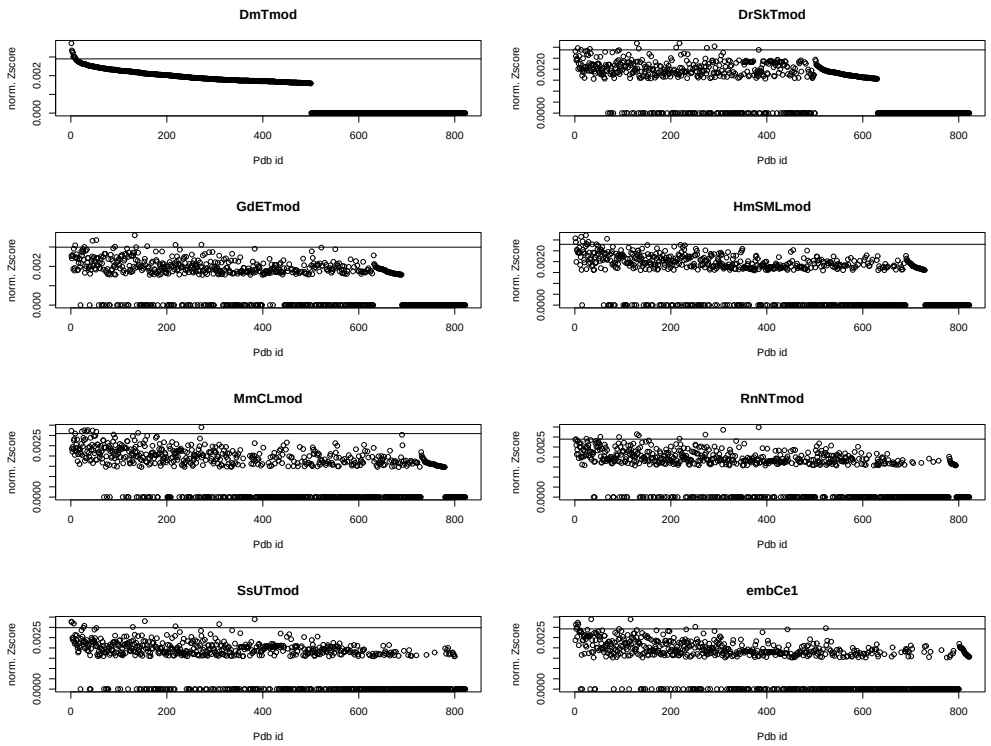

Figure 4: For each of the eight query sequences, the $-log(Evalues)$ are plotted against the indices of the top 500 hits. The horizontal line indicates the divider between the top 10 hits and the rest.

Figure 5: Initial model created by threading the GdETmod C-terminus onto the structure ofthe matching region from 1gky. This is the odd structure since it is differentfrom Cassie's other high ranking matches such as 1d0B and 1xum

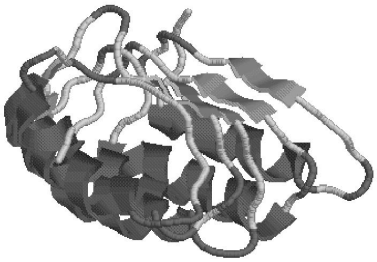

Figure 6: The starting model created by threading the GdETmod C-terminus onto the structure of the matching region from 2bnh. This is the predicted best starting model since it is similar to all the other top hits (except 1gky).

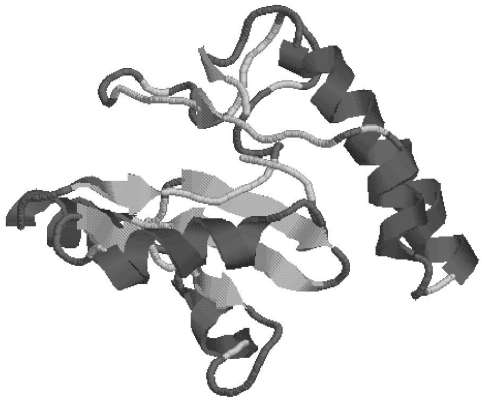

Figure 7: This figure shows the minimized structure for the C-terminus eTmod model, based on the structure of 1gky.

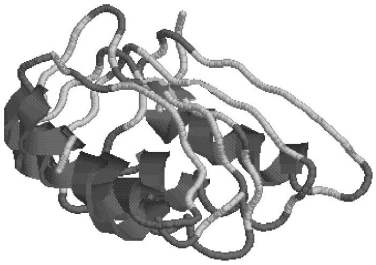

Figure 8: This figure shows the minimized structure for the C-terminus eTmod model, based on the structure of 2bnh.

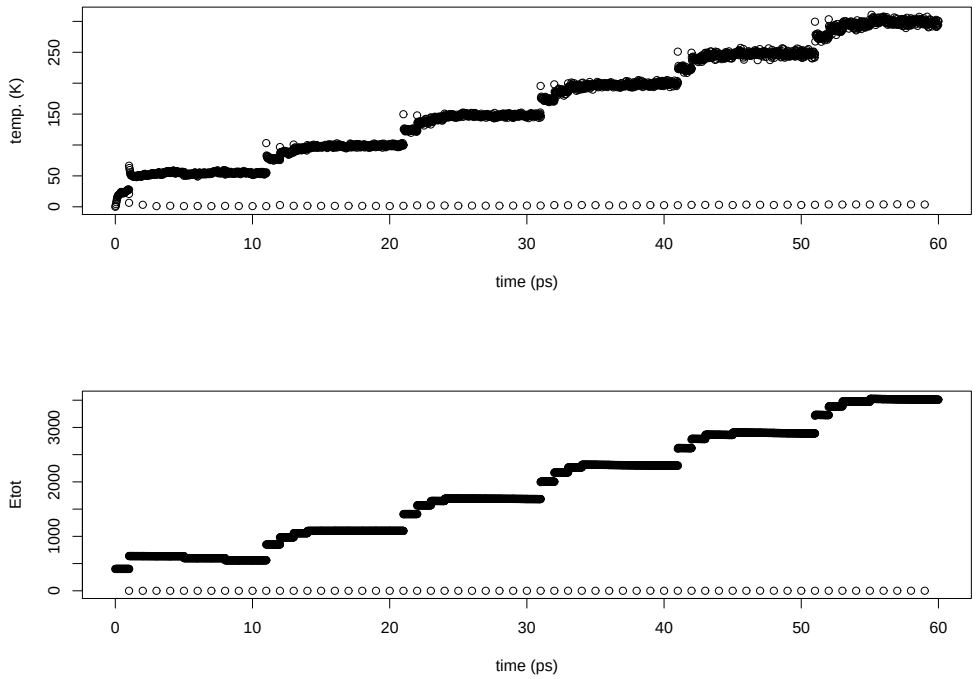

Figure 9: This figure shows the total energy and temperature during the heating portion of minimization. The idea is to slowly add heat to the protein and rescale each 10 picosecond.

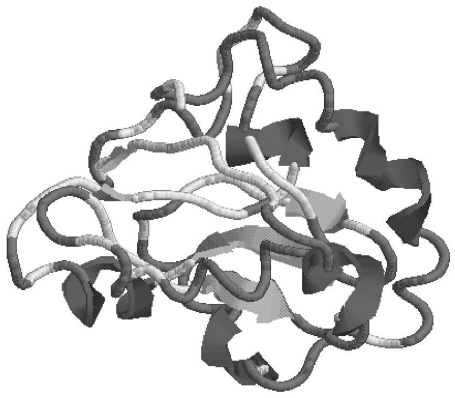

Figure 10: This figure shows the structure for the C-terminus eTmod model of 1gky, after heating it to 300K somewhat slowly, over 60 picoseconds.

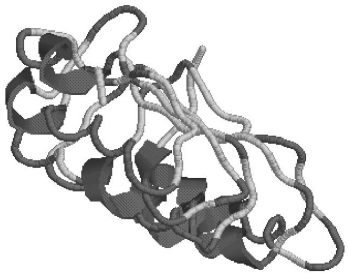

Figure 11: This figure shows the structure for the C-terminus eTmod model of 2bnh, after heating it to 300K somewhat slowly, over 60 picoseconds.

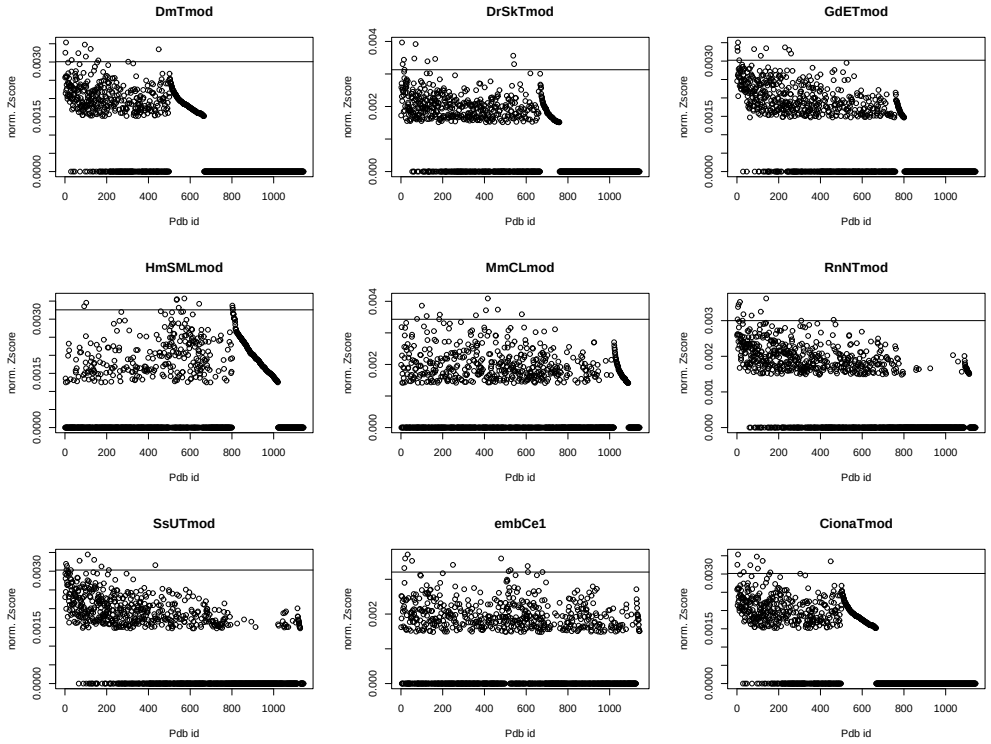

Figure 12: These are the compiled results for the top 500 hits from the UCLA/DOE fold server. A line divides the top 10 hits from the rest for each of the eight original sequences and CionaTmod.

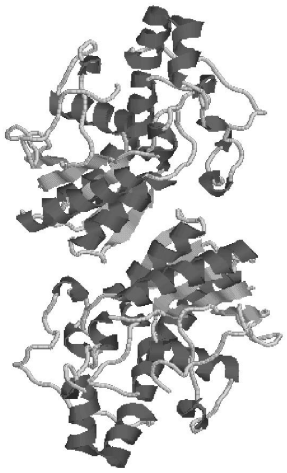

Figure 13: 1QHF

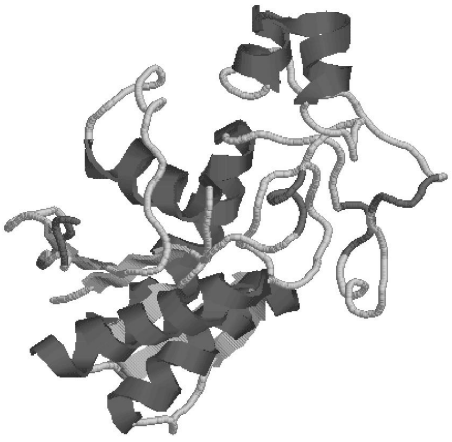

Figure 14: 3PGM

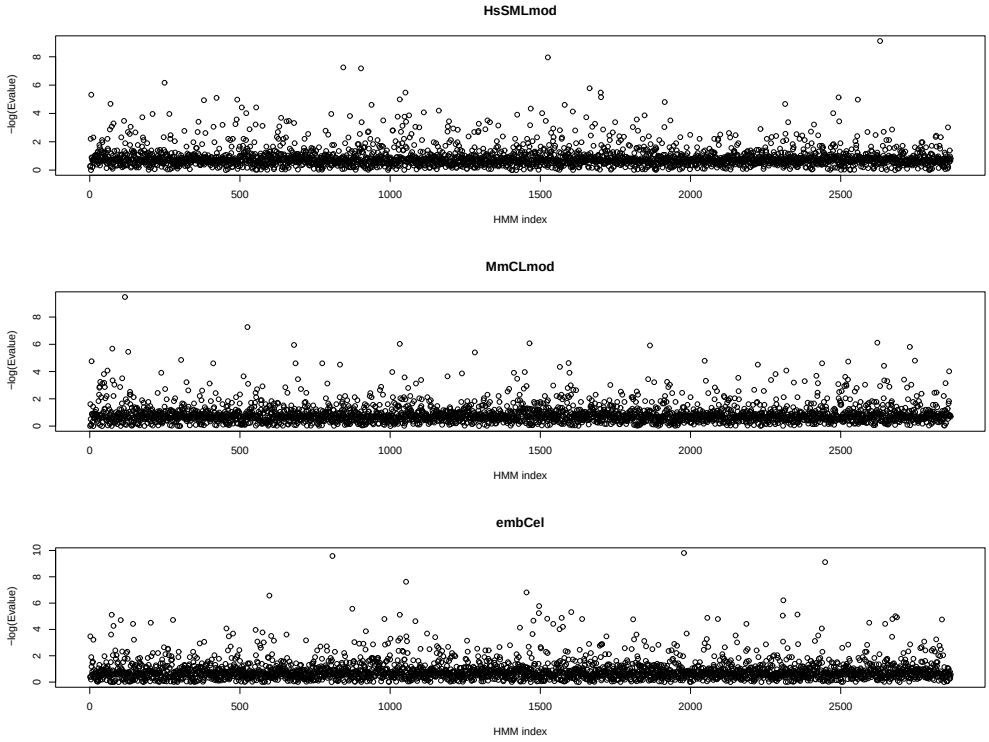

Figure 15: The $-log(Evalues)$ are compared for HsSMLmod, MmCLmod, and embCel. None of the hits are clearly higher than all the rest for all three sequences.